PORTAL
USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: The ACM Digital Library The Guide

free +link +memory

SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **free link**

Found **52,538** of **171,143**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Shared momory objects: Lock-free linked lists and skip lists

Mikhail Fomitchev, Eric Ruppert

July 2004 **Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(225.72 KB) Additional Information: full citation, abstract, references, index terms

Lock-free shared data structures implement distributed objects without the use of mutual exclusion, thus providing robustness and reliability. We present a new lock-free implementation of singly-linked lists. We prove that the worst-case amortized cost of the operations on our linked lists is linear in the length of the list plus the contention, which is better than in previous lock-free implementations of this data structure. Our implementation uses backlinks that are set when a node is deleted ...

Keywords: amortized analysis, analysis, distributed, efficient, fault-tolerant, linked list, lock-free, skip list

2 Lock-free linked lists using compare-and-swap

John D. Valois

August 1995 **Proceedings of the fourteenth annual ACM symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(902.29 KB) Additional Information: full citation, references, citings, index terms

3 A unified approach to loop-free routing using distance vectors or link states

J. J. Garcia-Luna-Aceves

August 1989 **ACM SIGCOMM Computer Communication Review , Symposium proceedings on Communications architectures & protocols SIGCOMM '89**, Volume 19 Issue 4

Publisher: ACM Press

Full text available: pdf(1.59 MB) Additional Information: full citation, abstract, references, citings, index terms

We present a unified approach for the dynamic computation of shortest paths in a computer network using either distance vectors or link states. We describe a distributed algorithm that provides loop-free paths at every instant and extends or improves algorithms introduced previously by Chandy and Misra, Jaffe and Moss, Merlin and Segall, and the author. Our approach treats the problem of distributed shortest-path routing as one of diffusing computations, which was first proposed by Dijkstra ...

4 Shadowed management of free disk pages with a linked list

Matthew S. Hecht, John D. Gabbe

December 1983 **ACM Transactions on Database Systems (TODS)**, Volume 8 Issue 4

Publisher: ACM Press

Full text available: pdf(877.39 KB) Additional Information: full citation, abstract, references, index terms

We describe and prove correct a programming technique using a linked list of pages for managing the free disk pages of a file system where shadowing is the recovery technique. Our technique requires a window of only two pages of main memory for accessing and maintaining the free list, and avoids wholesale copying of free-list pages during a checkpoint or recover operation.

Keywords: checkpoint, dynamic storage allocation, file system, recovery, shadowing, storage management

5 Loop-free routing using diffusing computations

J. J. Garcia-Lunes-Aceves

February 1993 **IEEE/ACM Transactions on Networking (TON)**, Volume 1 Issue 1

Publisher: IEEE Press

Full text available: pdf(1.38 MB) Additional Information: full citation, references, citings, index terms

6 Proof nets for unit-free multiplicative-additive linear logic

Dominic J. D. Hughes, Rob J. Van Glabbeek

October 2005 **ACM Transactions on Computational Logic (TOCL)**, Volume 6 Issue 4

Publisher: ACM Press

Full text available: pdf(1.14 MB) Additional Information: full citation, abstract, references, index terms

A cornerstone of the theory of proof nets for unit-free multiplicative linear logic (MLL) is the abstract representation of cut-free proofs modulo inessential rule commutation. The only known extension to additives, based on monomial weights, fails to preserve this key feature: a host of cut-free monomial proof nets can correspond to the same cut-free proof. Thus, the problem of finding a satisfactory notion of proof net for unit-free multiplicative-additive linear logic (MALL) has remained open ...

Keywords: Linear logic, additives, cut elimination, proof nets

7 Implementing wait-free objects on priority-based systems

James H. Anderson, Srikanth Ramamurthy, Rohit Jain

August 1997 **Proceedings of the sixteenth annual ACM symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(1.19 MB) Additional Information: full citation, references, citings, index terms

8 Garbage Collection of Linked Data Structures

Jacques Cohen

September 1981 **ACM Computing Surveys (CSUR)**, Volume 13 Issue 3

Publisher: ACM Press

Full text available: pdf(2.32 MB) Additional Information: full citation, references, citings, index terms

9 Minimum-link paths among obstacles in the plane

Joseph S. B. Mitchell, Günter Rote, Gerhard Woeginger

May 1990 **Proceedings of the sixth annual symposium on Computational geometry**

Publisher: ACM Press

Full text available: pdf(966.50 KB) Additional Information: full citation, abstract, references, citings, index terms

Given a set of nonintersecting polygonal obstacles in the plane, the link distance between two points s and t is the minimum number of edges required to form a polygonal path connecting s to t that avoids all obstacles. We present an algorithm that computes the link distance (and a corresponding minimum-link path) between two points in time &Ogr; (E&agr;(n) ...

10 A path-finding algorithm for loop-free routing

J. J. Garcia-Luna-Aceves, Shree Murthy

February 1997 **IEEE/ACM Transactions on Networking (TON)**, Volume 5 Issue 1

Publisher: IEEE Press

Full text available: pdf(414.16 KB) Additional Information: full citation, references, citings, index terms

Keywords: internetworking, loop freedom, routing, shortest path

11 Split-ordered lists: lock-free extensible hash tables

Ori Shalev, Nir Shavit

July 2003 **Proceedings of the twenty-second annual symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, citings, index terms

We present the first lock-free implementation of an extensible hash table running on current architectures. It provides concurrent insert, delete, and search operations with an expected *O*(1) cost. It consists of very simple code, easily implementable using only load, store, and compare-and-swap operations. The new mathematical structure at the core of our algorithm is *recursive split-ordering*, a way of ordering elements in a linked list so that they can be repeatedly "split" using ...

Keywords: Compare-and-Swap, Concurrent Data Structures, Hash Table, Non-blocking Synchronization, Real-Time

12 Linking by inking: trailblazing in a paper-like hypertext

Morgan N. Price, Gene Golovchinsky, Bill N. Schilit

May 1998 **Proceedings of the ninth ACM conference on Hypertext and hypermedia : links, objects, time and space---structure in hypermedia systems: links, objects, time and space---structure in hypermedia systems**

Publisher: ACM Press

Full text available: pdf(1.46 MB) Additional Information: full citation, references, citings, index terms

13 A method for implementing lock-free shared-data structures

Greg Barnes

August 1993 **Proceedings of the fifth annual ACM symposium on Parallel algorithms and architectures**

Publisher: ACM Press

Full text available: pdf(978.61 KB) Additional Information: full citation, references, citings, index terms

14 Session 3: High performance dynamic lock-free hash tables and list-based sets

Maged M. Michael
August 2002 **Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures**
Publisher: ACM Press

Full text available: pdf(238.11 KB) Additional Information: full citation, abstract, references, citings, index terms

Lock-free (non-blocking) shared data structures promise more robust performance and reliability than conventional lock-based implementations. However, all prior lock-free algorithms for sets and hash tables suffer from serious drawbacks that prevent or limit their use in practice. These drawbacks include size inflexibility, dependence on atomic primitives not supported on any current processor architecture, and dependence on highly-inefficient or blocking memory management techniques.Building on ...

15 Session 1: Safe memory reclamation for dynamic lock-free objects using atomic reads and writes
Maged M. Michael
July 2002 **Proceedings of the twenty-first annual symposium on Principles of distributed computing**
Publisher: ACM Press
Full text available: pdf(1.13 MB) Additional Information: full citation, abstract, references, citings

A major obstacle to the wide use of lock-free data structures, despite their many performance and reliability advantages, is the absence of a practical lock-free method for reclaiming the memory of dynamic nodes removed from dynamic lock-free objects for arbitrary reuse.The only prior lock-free memory reclamation method depends on the DCAS atomic primitive, which is not supported on any current processor architecture. Other memory management methods are blocking, require special operating system ...

16 Load balanced deadlock-free deterministic routing of arbitrary networks
David J. Pritchard
April 1992 **Proceedings of the 1992 ACM annual conference on Communications**
Publisher: ACM Press
Full text available: pdf(836.41 KB) Additional Information: full citation, abstract, references, index terms

This paper provides efficient algorithms to deadlock-free route arbitrary multiprocessor interconnection networks as follows: 1. An algorithm is derived for fixed directory routing on an arbitrary network topology such that messages will be routed via one of the shortest routes whilst maintaining an even distribution of traffic over the network (assuming that messages are generated and absorbed in an even manner, or two-phase random routing is used).

17 Link and channel measurement: A simple mechanism for capturing and replaying wireless channels
Glenn Judd, Peter Steenkiste
August 2005 **Proceeding of the 2005 ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis E-WIND '05**
Publisher: ACM Press
Full text available: pdf(6.06 MB) Additional Information: full citation, abstract, references, index terms

Physical layer wireless network emulation has the potential to be a powerful experimental tool. An important challenge in physical emulation, and traditional simulation, is to accurately model the wireless channel. In this paper we examine the possibility of using on-card signal strength measurements to capture wireless channel traces. A key advantage of this approach is the simplicity and ubiquity with which these measurements can be obtained since virtually all wireless devices provide the req ...

Keywords: channel capture, emulation, wireless

18
Fully-adaptive minimal deadlock-free packet routing in hypercubes, meshes, and

other networks
Gustavo D. Pifarré, Luis Gravano, Sergio A. Felperin, Jorge L. C. Sanz
June 1991 **Proceedings of the third annual ACM symposium on Parallel algorithms and architectures**
Publisher: ACM Press
Full text available: pdf(1.08 MB) Additional Information: full citation, references, citings, index terms

19 Link-time binary rewriting techniques for program compaction
Bjorn De Sutter, Bruno De Bus, Koen De Bosschere
September 2005 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 27 Issue 5
Publisher: ACM Press
Full text available: pdf(1.37 MB) Additional Information: full citation, abstract, references, index terms

Small program size is an important requirement for embedded systems with limited amounts of memory. We describe how link-time compaction through binary rewriting can achieve code size reductions of up to 62&percent; for statically bound languages such as C, C++, and Fortran, without compromising on performance. We demonstrate how the limited amount of information about a program at link time can be exploited to overcome overhead resulting from separate compilation. This is done with sc ...

Keywords: Program representation, binary rewriting, code abstraction, compaction, interprocedural analysis, linker, whole-program optimization

20 Channel access scheduling in Ad Hoc networks with unidirectional links
Lichun Bao, J. J. Garcia-Luna-Aceves
July 2001 **Proceedings of the 5th international workshop on Discrete algorithms and methods for mobile computing and communications**
Publisher: ACM Press
Full text available: pdf(796.06 KB) Additional Information: full citation, abstract, references, citings, index terms

A new family of collision-free channel access protocols for ad hoc networks with unidirectional links is introduced. These protocols are based on a distributed contention resolution algorithm that operates at each node based on the list of direct contenders (one-hop neighbors or incident links) and indirect interferences (two-hop neighbors and related links). Depending on the activation scheme (node activation or link activation), a network node uses the identifiers of its neighbors one and t ...

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2006 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

PORTAL
USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: The ACM Digital Library The Guide

+head +pointer +free +link

SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **head pointer free link**

Found **1,848** of **173,942**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Simple, fast, and practical non-blocking and blocking concurrent queue algorithms

Maged M. Michael, Michael L. Scott

May 1996 **Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(860.15 KB) Additional Information: full citation, references, citings, index terms

Keywords: compare_and_swap, concurrent queue, lock-free, multiprogramming, non-blocking

2 Shared momory objects: Lock-free linked lists and skip lists

Mikhail Fomitchev, Eric Ruppert

July 2004 **Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(225.72 KB) Additional Information: full citation, abstract, references, index terms

Lock-free shared data structures implement distributed objects without the use of mutual exclusion, thus providing robustness and reliability. We present a new lock-free implementation of singly-linked lists. We prove that the worst-case amortized cost of the operations on our linked lists is linear in the length of the list plus the contention, which is better than in previous lock-free implementations of this data structure. Our implementation uses backlinks that are set when a node is deleted ...

Keywords: amortized analysis, analysis, distributed, efficient, fault-tolerant, linked list, lock-free, skip list

3 Split-ordered lists: lock-free extensible hash tables

Ori Shalev, Nir Shavit

July 2003 **Proceedings of the twenty-second annual symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, citings, index terms

We present the first lock-free implementation of an extensible hash table running on current architectures. It provides concurrent insert, delete, and search operations with an expected $O(1)$ cost. It consists of very simple code, easily implementable using only load, store, and compare-and-swap operations. The new mathematical structure at the core of

our algorithm is *recursive split-ordering*, a way of ordering elements in a linked list so that they can be repeatedly "split" using ...

Keywords: Compare-and-Swap, Concurrent Data Structures, Hash Table, Non-blocking Synchronization, Real-Time

4 High-performance multi-queue buffers for VLSI communications switches

Y. Tamir, G. L. Frazier

May 1988 **ACM SIGARCH Computer Architecture News , Proceedings of the 15th Annual International Symposium on Computer architecture ISCA '88**, Volume 16 Issue 2

Publisher: IEEE Computer Society Press, ACM Press

Full text available: pdf(1.41 MB) Additional Information: full citation, abstract, references, citings, index terms

Small nxn switches are key components of multistage interconnection networks used in multiprocessors as well as in the communication coprocessors used in multicomputers. The design of the internal buffers in these switches is of critical importance for achieving high throughput low latency communication. We discuss several buffer structures and compare them in terms of implementation complexity and their ability to deal with variations in traffic patterns a ...

5 Managing Reentrant Structures Using Reference Counts

Daniel G. Bobrow

July 1980 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 2 Issue 3

Publisher: ACM Press

Full text available: pdf(317.55 KB) Additional Information: full citation, abstract, references, citings, index terms

Automatic storage management requires that one identify storage unreachable by a user's program and return it to free status. One technique maintains a count of the references from user's programs to each cell, since a count of zero implies the storage is unreachable. Reentrant structures are self-referencing; hence no cell in them will have a count of zero, even though the entire structure is unreachable. A modification of standard reference counting can be used to manaage the deallocation ...

6 Abstract description of pointer data structures: an approach for improving the analysis and optimization of imperative programs

Joseph Hummel, Laurie J. Hendren, Alexandru Nicolau

September 1992 **ACM Letters on Programming Languages and Systems (LOPLAS)**, Volume 1 Issue 3

Publisher: ACM Press

Full text available: pdf(1.23 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Even though impressive progress has been made in the area of optimizing and parallelizing array-based programs, the application of similar techniques to programs using pointer data structures has remained difficult. Unlike arrays which have a small number of well-defined properties, pointers can be used to implement a wide variety of structures which exhibit a much larger set of properties. The diversity of these structures implies that programs with pointer data structures cannot be effect ...

7 Memory system performance of UNIX on CC-NUMA multiprocessors

John Chapin, A. Herrod, Mendel Rosenblum, Anoop Gupta

May 1995 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1995 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems SIGMETRICS '95/PERFORMANCE '95**, Volume 23 Issue 1

Publisher: ACM Press

Additional Information:

Full text available: pdf(1.78 MB) full citation, abstract, references, citings, index terms

This study characterizes the performance of a variant of UNIX SVR4 on a large shared-memory multiprocessor and analyzes the effects of possible OS and architectural changes. We use a nonintrusive cache miss monitor to trace the execution of an OS-intensive multiprogrammed workload on the Stanford DASH, a 32-CPU CC-NUMA multiprocessor (CC-NUMA multiprocessors have cache-coherent shared memory that is physically distributed across the machine). We find that our version of UNIX accounts for 24% of ...

8 Session 3: High performance dynamic lock-free hash tables and list-based sets

Maged M. Michael
August 2002 **Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures**
Publisher: ACM Press

Full text available: pdf(238.11 KB) Additional Information: full citation, abstract, references, citings, index terms

Lock-free (non-blocking) shared data structures promise more robust performance and reliability than conventional lock-based implementations. However, all prior lock-free algorithms for sets and hash tables suffer from serious drawbacks that prevent or limit their use in practice. These drawbacks include size inflexibility, dependence on atomic primitives not supported on any current processor architecture, and dependence on highly-inefficient or blocking memory management techniques.Building on ...

9 On processes and interrupts

G. W. Gerrity
June 1981 **ACM SIGARCH Computer Architecture News**, Volume 9 Issue 4
Publisher: ACM Press
Full text available: pdf(566.33 KB) Additional Information: full citation, abstract, references

The key concept of a process in a multi-program, multi-processor operating system is examined in detail to highlight areas where appropriate hardware support can yield benefits. A sketch of one architecture providing appropriate support is given in sufficient detail to illustrate its flexibility and its low time and space overhead.

Keywords: hardware context switching, interrupts, operating system, process, scheduling

10 Design and evaluation of a DRAM-based shared memory ATM switch

Tzi-cker Chiueh, Srinidhi Varadarajan
June 1997 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1997 ACM SIGMETRICS international conference on Measurement and modeling of computer systems SIGMETRICS '97**, Volume 25 Issue 1
Publisher: ACM Press

Full text available: pdf(1.55 MB) Additional Information: full citation, abstract, references, citings, index terms

Beluga is a single-chip switch architecture specifically targeted at local area ATM networks, and it features three architectural innovations. First, an interconnection hierarchy composed of multiple switching fabrics is built into the chip to provide both low-latency cell transfer when the traffic is light and low cell drop rate under heavy load. Secondly, to improve silicon efficiency, Beluga is based on shared memory architecture, and the buffers are implemented using DRAM rather than ...

11 Design of the real-time executive for the Univac(r) 418 system

John Michael Williams
January 1966 **Proceedings of the 1966 21st national conference**
Publisher: ACM Press
Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, index terms

The UNIVAC 418 system hardware The 418 is a small- to medium-scale real-time computer announced to the general public in September of 1964. It is available in two models, identical except for storage speed (two or four microseconds). Storage and registers

12 Embedded hardware design case studies: A fully-programmable memory management system optimizing queue handling at multi gigabit rates

G. Kornaros, I. Papaefstathiou, A. Nikologiannis, N. Zervos
June 2003 **Proceedings of the 40th conference on Design automation**
Publisher: ACM Press
Full text available: pdf(236.13 KB) Additional Information: full citation, abstract, references, index terms

Two of the main bottlenecks when designing a network embedded system are very often the memory bandwidth and its capacity. This is mainly due to the extremely high speed of the state-of-the-art network links and to the fact that in order to support advanced quality of service (QoS), per-flow queueing is desirable. In this paper we describe the architecture of a memory manager that can provide up to 10Gbs of aggregate throughput while handling 512K queues. The presented system supports a complete ...

Keywords: memory management, network processor

13 Compile-time memory reuse in logic programming languages through update in place

Gudjón Gudjónsson, William H. Winsborough
May 1999 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 21 Issue 3
Publisher: ACM Press
Full text available: pdf(693.38 KB) Additional Information: full citation, abstract, references, index terms

Standard implementation techniques for single-assignment languages modify a data structure without destroying the original, which may subsequently be accessed. Instead a variant structure is created by using newly allocated cells to represent the changed portion and to replace any cell that references a newly allocated cell. The rest of the original structure is shared by the variant. The effort required to leave the original uncorrupted is unnecessary when the program will never reference ...

Keywords: Prolog, compile-time garbage collection, local reuse, reuse map, update in place

14 Memory-efficient state lookups with fast updates

Sandeep Sikka, George Varghese
August 2000 **ACM SIGCOMM Computer Communication Review , Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication SIGCOMM '00**, Volume 30 Issue 4
Publisher: ACM Press
Full text available: pdf(384.82 KB) Additional Information: full citation, abstract, references, citings, index terms

Routers must do a best matching prefix lookup for every packet; solutions for Gigabit speeds are well known. As Internet link speeds higher, we seek a scalable solution whose speed scales with memory speeds while allowing large prefix databases. In this paper we show that providing such a solution requires careful attention to memory allocation and pipelining. This is because fast lookups require on-chip or off-chip SRAM which is limited by either expense ...

15 Toward the efficient implementation of expert systems in Ada

S. Daniel Lee
December 1990 **Proceedings of the conference on TRI-ADA '90**
Publisher: ACM Press

Full text available: pdf(865.61 KB) Additional Information: full citation, abstract, references

Due to the Ada mandate of such government agencies as DoD, NASA and FAA, interest in deploying expert systems in Ada has increased. Recently, a couple of Ada-based expert system tools have been developed. According to recent benchmark reports, these tools do not perform as well as similar tools written in C. While poorly implemented Ada compilers also contribute to the poor benchmark result, some fundamental problems of the Ada language itself have been uncovered. In this paper, we describe ...

16 Current and future trends in compiler validation and evaluation (invited presentation)

John Solomond
December 1990 **Proceedings of the conference on TRI-ADA '90**
Publisher: ACM Press
Full text available: pdf(865.61 KB) Additional Information: full citation, references

17 Memory safety without runtime checks or garbage collection

Dinakar Dhurjati, Sumant Kowshik, Vikram Adve, Chris Lattner
June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems LCTES '03**, Volume 38 Issue 7
Publisher: ACM Press
Full text available: pdf(245.47 KB) Additional Information: full citation, abstract, references, citings, index terms

Traditional approaches to enforcing memory safety of programs rely heavily on runtime checks of memory accesses and on garbage collection, both of which are unattractive for embedded applications. The long-term goal of our work is to enable 100% static enforcement of memory safety for embedded programs through advanced compiler techniques and minimal semantic restrictions on programs. The key result of this paper is a compiler technique that ensures memory safety of dynamically allocated memory ...

Keywords: automatic pool allocation, compilers, embedded systems, programming languages, region management, security, static analysis

18 Error checking with pointer variables

Marvin V. Zelkowitz, Paul R. McMullin, Keith R. Merkel, Howard J. Larsen
October 1976 **Proceedings of the annual conference**
Publisher: ACM Press
Full text available: pdf(518.24 KB) Additional Information: full citation, abstract, references, index terms

The use of pointer variables in a programming language often results in a difficult class of errors to detect. Pointers may point to storage that no longer is allocated, or storage may be allocated as one data type and accessed as another. This report describes the implementation of pointers in the PLUM PL/1 compiler such that all error conditions are detected. In addition, preliminary tests with PLUM seem to indicate that the total checking of pointers is not as expensive an operation as i ...

19 Performance analysis of on-the-fly garbage collection

Tim Hickey, Jacques Cohen
November 1984 **Communications of the ACM**, Volume 27 Issue 11
Publisher: ACM Press
Full text available: pdf(828.76 KB) Additional Information: full citation, references, citings, index terms

Keywords: efficiency, list processing, marking algorithms, parallel garbage collection, speedup



20 Diffracting trees



Nir Shavit, Asaph Zemach
November 1996 **ACM Transactions on Computer Systems (TOCS)**, Volume 14 Issue 4
Publisher: ACM Press

Full text available: pdf(729.57 KB) Additional Information: full citation, abstract, references, citings, index terms

Shared counters are among the most basic coordination structures in multiprocessor computation, with applications ranging from barrier synchronization to concurrent-data-structure design. This article introduces diffracting trees, novel data structures for share counting and load balancing in a distributed/parallel environment. Empirical evidence, collected on a simulated distributed shared-memory machine and several simulated message-passing architectures, shows that diffracting trees scal ...

Keywords: contention, counting networks, index distribution, lock free, wait free

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2006 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

PORTAL
USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: The ACM Digital Library The Guide

+head +pointer +free +link

SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **head pointer free link**

Found **1,848** of **173,942**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Simple, fast, and practical non-blocking and blocking concurrent queue algorithms

Maged M. Michael, Michael L. Scott

May 1996 **Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(860.15 KB) Additional Information: full citation, references, citings, index terms

Keywords: compare_and_swap, concurrent queue, lock-free, multiprogramming, non-blocking

2 Shared momory objects: Lock-free linked lists and skip lists

Mikhail Fomitchev, Eric Ruppert

July 2004 **Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(225.72 KB) Additional Information: full citation, abstract, references, index terms

Lock-free shared data structures implement distributed objects without the use of mutual exclusion, thus providing robustness and reliability. We present a new lock-free implementation of singly-linked lists. We prove that the worst-case amortized cost of the operations on our linked lists is linear in the length of the list plus the contention, which is better than in previous lock-free implementations of this data structure. Our implementation uses backlinks that are set when a node is deleted ...

Keywords: amortized analysis, analysis, distributed, efficient, fault-tolerant, linked list, lock-free, skip list

3 Split-ordered lists: lock-free extensible hash tables

Ori Shalev, Nir Shavit

July 2003 **Proceedings of the twenty-second annual symposium on Principles of distributed computing**

Publisher: ACM Press

Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, citings, index terms

We present the first lock-free implementation of an extensible hash table running on current architectures. It provides concurrent insert, delete, and search operations with an expected $O(1)$ cost. It consists of very simple code, easily implementable using only load, store, and compare-and-swap operations. The new mathematical structure at the core of

our algorithm is *recursive split-ordering*, a way of ordering elements in a linked list so that they can be repeatedly "split" using ...

Keywords: Compare-and-Swap, Concurrent Data Structures, Hash Table, Non-blocking Synchronization, Real-Time

4 High-performance multi-queue buffers for VLSI communications switches

Y. Tamir, G. L. Frazier

May 1988 **ACM SIGARCH Computer Architecture News , Proceedings of the 15th Annual International Symposium on Computer architecture ISCA '88**, Volume 16 Issue 2

Publisher: IEEE Computer Society Press, ACM Press

Full text available: pdf(1.41 MB) Additional Information: full citation, abstract, references, citings, index terms

Small nxn switches are key components of multistage interconnection networks used in multiprocessors as well as in the communication coprocessors used in multicomputers. The design of the internal buffers in these switches is of critical importance for achieving high throughput low latency communication. We discuss several buffer structures and compare them in terms of implementation complexity and their ability to deal with variations in traffic patterns a ...

5 Managing Reentrant Structures Using Reference Counts

Daniel G. Bobrow

July 1980 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 2 Issue 3

Publisher: ACM Press

Full text available: pdf(317.55 KB) Additional Information: full citation, abstract, references, citings, index terms

Automatic storage management requires that one identify storage unreachable by a user's program and return it to free status. One technique maintains a count of the references from user's programs to each cell, since a count of zero implies the storage is unreachable. Reentrant structures are self-referencing; hence no cell in them will have a count of zero, even though the entire structure is unreachable. A modification of standard reference counting can be used to manaage the deallocation ...

6 Abstract description of pointer data structures: an approach for improving the analysis and optimization of imperative programs

Joseph Hummel, Laurie J. Hendren, Alexandru Nicolau

September 1992 **ACM Letters on Programming Languages and Systems (LOPLAS)**, Volume 1 Issue 3

Publisher: ACM Press

Full text available: pdf(1.23 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Even though impressive progress has been made in the area of optimizing and parallelizing array-based programs, the application of similar techniques to programs using pointer data structures has remained difficult. Unlike arrays which have a small number of well-defined properties, pointers can be used to implement a wide variety of structures which exhibit a much larger set of properties. The diversity of these structures implies that programs with pointer data structures cannot be effect ...

7 Memory system performance of UNIX on CC-NUMA multiprocessors

John Chapin, A. Herrod, Mendel Rosenblum, Anoop Gupta

May 1995 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1995 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems SIGMETRICS '95/PERFORMANCE '95**, Volume 23 Issue 1

Publisher: ACM Press

Additional Information:

Full text available: pdf(1.78 MB) full citation, abstract, references, citings, index terms

This study characterizes the performance of a variant of UNIX SVR4 on a large shared-memory multiprocessor and analyzes the effects of possible OS and architectural changes. We use a nonintrusive cache miss monitor to trace the execution of an OS-intensive multiprogrammed workload on the Stanford DASH, a 32-CPU CC-NUMA multiprocessor (CC-NUMA multiprocessors have cache-coherent shared memory that is physically distributed across the machine). We find that our version of UNIX accounts for 24% of ...

8 Session 3: High performance dynamic lock-free hash tables and list-based sets

Maged M. Michael

August 2002 **Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures**

Publisher: ACM Press

Full text available: pdf(238.11 KB) Additional Information: full citation, abstract, references, citings, index terms

Lock-free (non-blocking) shared data structures promise more robust performance and reliability than conventional lock-based implementations. However, all prior lock-free algorithms for sets and hash tables suffer from serious drawbacks that prevent or limit their use in practice. These drawbacks include size inflexibility, dependence on atomic primitives not supported on any current processor architecture, and dependence on highly-inefficient or blocking memory management techniques.Building on ...

9 On processes and interrupts

G. W. Gerrity

June 1981 **ACM SIGARCH Computer Architecture News**, Volume 9 Issue 4

Publisher: ACM Press

Full text available: pdf(566.33 KB) Additional Information: full citation, abstract, references

The key concept of a process in a multi-program, multi-processor operating system is examined in detail to highlight areas where appropriate hardware support can yield benefits. A sketch of one architecture providing appropriate support is given in sufficient detail to illustrate its flexibility and its low time and space overhead.

Keywords: hardware context switching, interrupts, operating system, process, scheduling

10 Design and evaluation of a DRAM-based shared memory ATM switch

Tzi-cker Chiueh, Srinidhi Varadarajan

June 1997 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1997 ACM SIGMETRICS international conference on Measurement and modeling of computer systems SIGMETRICS '97**, Volume 25 Issue 1

Publisher: ACM Press

Full text available: pdf(1.55 MB) Additional Information: full citation, abstract, references, citings, index terms

Beluga is a single-chip switch architecture specifically targeted at local area ATM networks, and it features three architectural innovations. First, an interconnection hierarchy composed of multiple switching fabrics is built into the chip to provide both low-latency cell transfer when the traffic is light and low cell drop rate under heavy load. Secondly, to improve silicon efficiency, Beluga is based on shared memory architecture, and the buffers are implemented using DRAM rather than ...

11 Design of the real-time executive for the Univac(r) 418 system

John Michael Williams

January 1966 **Proceedings of the 1966 21st national conference**

Publisher: ACM Press

Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, index terms

The UNIVAC 418 system hardware The 418 is a small- to medium-scale real-time computer announced to the general public in September of 1964. It is available in two models, identical except for storage speed (two or four microseconds). Storage and registers

12 Embedded hardware design case studies: A fully-programmable memory management system optimizing queue handling at multi gigabit rates

G. Kornaros, I. Papaefstathiou, A. Nikologiannis, N. Zervos

June 2003 **Proceedings of the 40th conference on Design automation**

Publisher: ACM Press

Full text available: pdf(236.13 KB) Additional Information: full citation, abstract, references, index terms

Two of the main bottlenecks when designing a network embedded system are very often the memory bandwidth and its capacity. This is mainly due to the extremely high speed of the state-of-the-art network links and to the fact that in order to support advanced quality of service (QoS), per-flow queueing is desirable. In this paper we describe the architecture of a memory manager that can provide up to 10Gbs of aggregate throughput while handling 512K queues. The presented system supports a complete ...

Keywords: memory management, network processor

13 Compile-time memory reuse in logic programming languages through update in place

Gudjón Gudjónsson, William H. Winsborough

May 1999 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 21 Issue 3

Publisher: ACM Press

Full text available: pdf(693.38 KB) Additional Information: full citation, abstract, references, index terms

Standard implementation techniques for single-assignment languages modify a data structure without destroying the original, which may subsequently be accessed. Instead a variant structure is created by using newly allocated cells to represent the changed portion and to replace any cell that references a newly allocated cell. The rest of the original structure is shared by the variant. The effort required to leave the original uncorrupted is unnecessary when the program will never reference ...

Keywords: Prolog, compile-time garbage collection, local reuse, reuse map, update in place

14 Memory-efficient state lookups with fast updates

Sandeep Sikka, George Varghese

August 2000 **ACM SIGCOMM Computer Communication Review , Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication SIGCOMM '00**, Volume 30 Issue 4

Publisher: ACM Press

Full text available: pdf(384.82 KB) Additional Information: full citation, abstract, references, citings, index terms

Routers must do a best matching prefix lookup for every packet; solutions for Gigabit speeds are well known. As Internet link speeds higher, we seek a scalable solution whose speed scales with memory speeds while allowing large prefix databases. In this paper we show that providing such a solution requires careful attention to memory allocation and pipelining. This is because fast lookups require on-chip or off-chip SRAM which is limited by either expense ...

15 Toward the efficient implementation of expert systems in Ada

S. Daniel Lee

December 1990 **Proceedings of the conference on TRI-ADA '90**

Publisher: ACM Press

Full text available: pdf(865.61 KB) Additional Information: full citation, abstract, references

Due to the Ada mandate of such government agencies as DoD, NASA and FAA, interest in deploying expert systems in Ada has increased. Recently, a couple of Ada-based expert system tools have been developed. According to recent benchmark reports, these tools do not perform as well as similar tools written in C. While poorly implemented Ada compilers also contribute to the poor benchmark result, some fundamental problems of the Ada language itself have been uncovered. In this paper, we describe ...

16 Current and future trends in compiler validation and evaluation (invited presentation)

John Solomond
December 1990 **Proceedings of the conference on TRI-ADA '90**
Publisher: ACM Press
Full text available: pdf(865.61 KB) Additional Information: full citation, references

17 Memory safety without runtime checks or garbage collection

Dinakar Dhurjati, Sumant Kowshik, Vikram Adve, Chris Lattner
June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems LCTES '03**, Volume 38 Issue 7
Publisher: ACM Press
Full text available: pdf(245.47 KB) Additional Information: full citation, abstract, references, citings, index terms

Traditional approaches to enforcing memory safety of programs rely heavily on runtime checks of memory accesses and on garbage collection, both of which are unattractive for embedded applications. The long-term goal of our work is to enable 100% static enforcement of memory safety for embedded programs through advanced compiler techniques and minimal semantic restrictions on programs. The key result of this paper is a compiler technique that ensures memory safety of dynamically allocated memory ...

Keywords: automatic pool allocation, compilers, embedded systems, programming languages, region management, security, static analysis

18 Error checking with pointer variables

Marvin V. Zelkowitz, Paul R. McMullin, Keith R. Merkel, Howard J. Larsen
October 1976 **Proceedings of the annual conference**
Publisher: ACM Press
Full text available: pdf(518.24 KB) Additional Information: full citation, abstract, references, index terms

The use of pointer variables in a programming language often results in a difficult class of errors to detect. Pointers may point to storage that no longer is allocated, or storage may be allocated as one data type and accessed as another. This report describes the implementation of pointers in the PLUM PL/1 compiler such that all error conditions are detected. In addition, preliminary tests with PLUM seem to indicate that the total checking of pointers is not as expensive an operation as i ...

19 Performance analysis of on-the-fly garbage collection

Tim Hickey, Jacques Cohen
November 1984 **Communications of the ACM**, Volume 27 Issue 11
Publisher: ACM Press
Full text available: pdf(828.76 KB) Additional Information: full citation, references, citings, index terms

Keywords: efficiency, list processing, marking algorithms, parallel garbage collection, speedup



20 Diffracting trees



Nir Shavit, Asaph Zemach
November 1996 **ACM Transactions on Computer Systems (TOCS)**, Volume 14 Issue 4
Publisher: ACM Press

Full text available: pdf(729.57 KB) Additional Information: full citation, abstract, references, citings, index terms

Shared counters are among the most basic coordination structures in multiprocessor conputation, with applications ranging from barrier synchronization to concurrent-data-structure design. This article introduces diffracting trees, novel data structures for share counting and load balancing in a distributed/parallel environment. Empirical evidence, collected on a simulated distributed shared-memory machine and several simulated message-passing architectures, shows that diffracting trees scal ...

Keywords: contention, counting networks, index distribution, lock free, wait free

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2006 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S8	20	((storage memory)) same (creat$4 mak$4 build$4 generat$4 construct$4) with (link list) same (identif$4 detect$4 determin$4 check$4) with (free empty) same (memory location) and((splic$4 merg$4 join$4 unit$4 link$4) with (memory location)) and (pointer) and (tail same head) and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/10 12:22
S10	5	((storage memory)) same (creat$4 mak$4 build$4 generat$4 construct$4) with (link list) same (identif$4 detect$4 determin$4 check$4) with (free empty) same (memory location) and((splic$4 merg$4 join$4 unit$4 link$4) with (memory location)) and (pointer) and (tail same (head with plus)) and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/10 12:23
S15	6	(splic$5 join$4 connect$4) with memory same (location address) same free with link with list and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/03/28 12:12
S18	7	(splic$5 join$4 connect$4) same memory same (location address) same free with link with list and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/03/28 12:13
S20	17	(splic$5 join$4 connect$4) same head same tail same pointer and "711"/$.ccls. and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/03/28 12:15
S21	1	(splic$5 join$4 connect$4) same head same tail same pointer and (free with link with list) and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/03/28 12:14

EAST Search History

S22	141	head same tail same pointer same link with list and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/03/28 12:16
S23	6	(allocat$4 assign$4) same (location address) same head same pointer same (free with link)and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/03/28 12:20
S24	89	(head with pointer) same free same link with list and (@ad<"20011221" @prad<"20011221" @rlad<"20011221")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/03/28 12:44

```
? d s
Set     Items   Description
S1      18971   S SPLICE? OR SPLICING?
S2    1160156   S BIND? ? OR BINDING OR INCORPORAT? OR COMBINE? OR COMBINING? OR MERGE? OR
MERGING? OR AMALGAM?
S3     191591   S BLEND? OR (ADD OR ADDS OR ADDING OR ADDED OR ADDITION)()TOGETHER?
S4      10574   S COMMINGL? OR CONJOIN? OR MINGL? OR INTERMIX? OR INTERMINGL? OR COMMIX?
S5     810693   S FUSE? OR FUSING? OR MELD? OR UNITE? OR UNITING? OR INTERBLEND? OR
INTEGRAT? OR UNIFY? OR UNIFIC?
S6    1296800   S INTERWEAV? OR INTERWOVEN? OR JOIN? OR LINK? OR TACK???()ON
S7    1930266   S MEMOR? OR STORAG? OR ROM? ? OR PROM? ? OR EPROM? ? OR EEPROM? ?
S8     482742   S RAM? OR DRAM? ? OR VRAM? OR NVRAM? OR SRAM? ? OR CACHE? OR BUFFER?
S9    2007363   S STORE? OR STORING? OR STORAG?
S10    377530   S S1:S6 AND S7:S9
S11     16525   S SUFFICIEN? OR AVAILAB? OR SPARE? OR UNRESERV? OR EXTRANEOUS?
S12      3857   S UNUSED? OR "NOT" ()USED OR VACANT? OR UNWRITTEN? OR USABL? OR SURPLUS?
OR SUPERFLUOUS?
S13     30468   S OPEN OR EXTRA OR FREE
S14       871   S SMALLEST OR SHORTEST OR LITTLEST OR TINIEST OR PETITEST
S15         2   S MOST()(SMALL OR SHORT OR LITTLE OR TINY OR PETITE? OR DIMINUIT? OR
TRUNCAT?)
S16         1   S MOST()(ABBREVIAT? OR ABRIDG? OR CONDENS?)
S17       756   S (LEAST OR LOWEST)(2N)(LARGE OR GREAT OR BIG OR SIZE OR AMOUNT? OR LIMIT?
OR SPACE?)
S18       615   S (LEAST OR LOWEST)(2N)(DIMENSION? OR LENGTH? OR CAPACIT? OR ROOM? ?)
S19     29179   S LOCATION? OR LIST? ? OR DIRECTORY? OR DIRECTORIE? OR TABLE?
S20     38667   S ROW? ? OR COLUMN? ? OR ADDRESS? OR ENTRY? OR ENTRIE?
S21        10   S (HEAD? ?(2N)TAIL? ?)(5N)POINTER?
S22     73237   S IC=G06F?
S23     62520   S MC=T01?
S24      2347   S S10 AND S1:S6(7N)S11:S17(7N)S7:S9
S25         1   S S24 AND S11:S13(7N)S14:S18(7N)S19:S21
S26       130   S S10 AND S1:S6 AND S11:S13 AND S14:S18 AND S19:S21
S27       130   S S25:S26
S28        37   S S27 AND S22:S23
S29       130   S S27:S28
S30       130   S S25:S29
S31        75   S S30 AND AC=US/PR
S32        50   S31 AND AY=(1970:2001)/PR FROM 347, 350
S33        50   S S31 AND AY=(1970:2001)/PR
S34        38   S S31 NOT AY=(2002:2006)/PR
S35        55   S S30 NOT S31
S36        38   S S35 AND PY=1970:2001
S37        32   S S35 NOT PY=2002:2006
S38        88   S S33:S34 OR S36:S37
S39        88   IDPAT (sorted in duplicate/non-duplicate order)
 ;' show files
```

[File 347] **JAPIO** Nov 1976-2005/Nov(Updated 060302)
(c) 2006 JPO & JAPIO. All rights reserved.

[File 350] **Derwent WPIX** 1963-2006/UD,UM &UP=200620
(c) 2006 Thomson Derwent. All rights reserved.
**File 350: For more current information, include File 331 in your search. Enter HELP NEWS 331 for details.*

```
? d s
Set     Items   Description
S1      76963   S SPLICE? OR SPLICING?
S2    4259841   S BIND? ? OR BINDING OR INCORPORAT? OR COMBINE? OR COMBINING? OR MERGE? OR
MERGING? OR AMALGAM?
S3     225526   S BLEND? OR (ADD OR ADDS OR ADDING OR ADDED OR ADDITION)()TOGETHER?
S4      41774   S COMMINGL? OR CONJOIN? OR MINGL? OR INTERMIX? OR INTERMINGL? OR COMMIX?
S5    5583170   S FUSE? OR FUSING? OR MELD? OR UNITE? OR UNITING? OR INTERBLEND? OR
INTEGRAT? OR UNIFY? OR UNIFIC?
S6    2482451   S INTERWEAV? OR INTERWOVEN? OR JOIN? OR LINK? OR TACK???()ON
S7    1760670   S MEMOR? OR STORAG? OR ROM? ? OR PROM? ? OR EPROM? ? OR EEPROM? ?
S8     971776   S RAM? OR DRAM? ? OR VRAM? OR NVRAM? OR SRAM? ? OR CACHE? OR BUFFER?
S9    1693032   S STORE? OR STORING? OR STORAG? OR UPDAT? OR BACKUP? OR BACK?()UP
S10    674720   S S1:S6 AND S7:S9
S11     58891   S SUFFICIEN? OR ENOUGH? OR AMPLE? OR AVAILAB? OR SPARE? OR UNRESERV? OR
EXTRANEOUS?
S12      2811   S UNUSED? OR "NOT" ()USED OR VACANT? OR UNWRITTEN? OR USABL? OR SURPLUS?
OR SUPERFLUOUS?
S13     45066   S OPEN OR EXTRA OR FREE
S14      2873   S SMALLEST OR SHORTEST OR LITTLEST OR TINIEST OR PETITEST
S15        22   S MOST()(SMALL OR SHORT OR LITTLE OR TINY OR PETITE? OR DIMINUIT? OR
TRUNCAT?)
S16         4   S MOST()(ABBREVIAT? OR ABRIDG? OR CONDENS?)
S17         1   S LEAST()(LARGE OR GREAT OR BIG OR SIZE)
S18        52   S LEAST(2N)(DIMENSION? OR LENGTH?)
S19     35966   S LOCATION? OR LIST? ? OR DIRECTORY? OR DIRECTORIE? OR TABLE?
S20     48242   S ROW? ? OR COLUMN? OR ADDRESS? OR ENTRY? OR ENTRIE?
S21         3   S (HEAD?(3N)TAIL?)(5N)POINTER?
S22         4   S S10 AND S1:S6(7N)S11:S13(7N)S7:S8 AND S14:S18(7N)S19:S21
S23       229   S S10 AND S11:S13 AND S14:S18 AND S19:S21
S24       124   S S23 AND S1:S6(10N)(S7:S9 OR S11:S21)
S25       229   S S22:S24
S26       140   S S25 AND PY<2002
S27       140   S S25 NOT PY>2001
S28       140   S S26:S27
S29       107   RD  (unique items)
 ; show files
```

[File 2] **INSPEC** 1898-2006/Mar W3
(c) 2006 Institution of Electrical Engineers. All rights reserved.

[File 6] **NTIS** 1964-2006/Mar W2
(c) 2006 NTIS, Intl Cpyrght All Rights Res. All rights reserved.

[File 8] **Ei Compendex(R)** 1970-2006/Mar W3
(c) 2006 Elsevier Eng. Info. Inc. All rights reserved.

[File 34] **SciSearch(R) Cited Ref Sci** 1990-2006/Mar W3
(c) 2006 Inst for Sci Info. All rights reserved.

[File 35] **Dissertation Abs Online** 1861-2006/Mar
(c) 2006 ProQuest Info&Learning. All rights reserved.

[File 65] **Inside Conferences** 1993-2006/Mar 28
(c) 2006 BLDSC all rts. reserv. All rights reserved.

[File 94] **JICST-EPlus** 1985-2006/Jan W1

(c)2006 Japan Science and Tech Corp(JST). All rights reserved.

[File 99] **Wilson Appl. Sci & Tech Abs** 1983-2006/Feb
(c) 2006 The HW Wilson Co. All rights reserved.

[File 111] **TGG Natl.Newspaper Index(SM)** 1979-2006/Mar 21
(c) 2006 The Gale Group. All rights reserved.

[File 144] **Pascal** 1973-2006/Mar W1
(c) 2006 INIST/CNRS. All rights reserved.

[File 239] **Mathsci** 1940-2006/May
(c) 2006 American Mathematical Society. All rights reserved.

[File 256] **TecInfoSource** 82-2006/Apr
(c) 2006 Info.Sources Inc. All rights reserved.